REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
05-03-2012	Final	1 January 2008 - 30 November 2011

4. TITLE AND SUBTITLE
(YIP-08) Automated, Certified Program-rewriting for Software Security Enforcement

5a. CONTRACT NUMBER FA9550-08-1-0044

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Hamlen, Kevin W.

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
The University of Texas at Dallas
800 W. Campbell Rd.
Richardson, TX 75080-3021

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Office of Scientific Research
875 North Randolph Street
Suite 325, Rm 3112
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S) AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-OSR-VA-TR-2012-0496

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approve for Public Release

13. SUPPLEMENTARY NOTES
Year 4 of the project finalized, tested, and published the Chekov IRM verification system (see outcome 2 of attached report), and extended the Reins SFI system to Linux-based architectures (see outcome 3 of attached report).

14. ABSTRACT
This project discovered and developed algorithms and tools for (1) automatically retrofitting binary legacy software with access controls, and (2) formally machine-certifying that the retrofitted software satisfies user-specified security policies. The research resulted in new software security systems for Java, ActionScript, and x86 native code that provably secure legacy code without any form of code-producer cooperation (e.g., source code or compiler support).

15. SUBJECT TERMS
software security, validation, runtime monitors, access controls

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					Kevin W. Hamlen
					19b. TELEPHONE NUMBER *(Include area code)*
U	U	U	UU	10	(972) 883-4724

INSTRUCTIONS FOR COMPLETING SF 298

1. REPORT DATE. Full publication date, including day, month, if available. Must cite at least the year and be Year 2000 compliant, e.g. 30-06-1998; xx-06-1998; xx-xx-1998.

2. REPORT TYPE. State the type of report, such as final, technical, interim, memorandum, master's thesis, progress, quarterly, research, special, group study, etc.

3. DATES COVERED. Indicate the time during which the work was performed and the report was written, e.g., Jun 1997 - Jun 1998; 1-10 Jun 1996; May - Nov 1998; Nov 1998.

4. TITLE. Enter title and subtitle with volume number and part number, if applicable. On classified documents, enter the title classification in parentheses.

5a. CONTRACT NUMBER. Enter all contract numbers as they appear in the report, e.g. F33615-86-C-5169.

5b. GRANT NUMBER. Enter all grant numbers as they appear in the report, e.g. AFOSR-82-1234.

5c. PROGRAM ELEMENT NUMBER. Enter all program element numbers as they appear in the report, e.g. 61101A.

5d. PROJECT NUMBER. Enter all project numbers as they appear in the report, e.g. 1F665702D1257; ILIR.

5e. TASK NUMBER. Enter all task numbers as they appear in the report, e.g. 05; RF0330201; T4112.

5f. WORK UNIT NUMBER. Enter all work unit numbers as they appear in the report, e.g. 001; AFAPL30480105.

6. AUTHOR(S). Enter name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. The form of entry is the last name, first name, middle initial, and additional qualifiers separated by commas, e.g. Smith, Richard, J, Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES). Self-explanatory.

8. PERFORMING ORGANIZATION REPORT NUMBER. Enter all unique alphanumeric report numbers assigned by the performing organization, e.g. BRL-1234; AFWL-TR-85-4017-Vol-21-PT-2.

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES). Enter the name and address of the organization(s) financially responsible for and monitoring the work.

10. SPONSOR/MONITOR'S ACRONYM(S). Enter, if available, e.g. BRL, ARDEC, NADC.

11. SPONSOR/MONITOR'S REPORT NUMBER(S). Enter report number as assigned by the sponsoring/ monitoring agency, if available, e.g. BRL-TR-829; -215.

12. DISTRIBUTION/AVAILABILITY STATEMENT. Use agency-mandated availability statements to indicate the public availability or distribution limitations of the report. If additional limitations/ restrictions or special markings are indicated, follow agency authorization procedures, e.g. RD/FRD, PROPIN, ITAR, etc. Include copyright information.

13. SUPPLEMENTARY NOTES. Enter information not included elsewhere such as: prepared in cooperation with; translation of; report supersedes; old edition number, etc.

14. ABSTRACT. A brief (approximately 200 words) factual summary of the most significant information.

15. SUBJECT TERMS. Key words or phrases identifying major concepts in the report.

16. SECURITY CLASSIFICATION. Enter security classification in accordance with security classification regulations, e.g. U, C, S, etc. If this form contains classified information, stamp classification level on the top and bottom of this page.

17. LIMITATION OF ABSTRACT. This block must be completed to assign a distribution limitation to the abstract. Enter UU (Unclassified Unlimited) or SAR (Same as Report). An entry in this block is necessary if the abstract is to be limited.

Final Report: YIP-08

Automated, Certified Program-rewriting for Software Security Enforcement

Grant/Contract Number: FA9550-08-1-0044

Kevin W. Hamlen

5 March 2012

Abstract

This project discovered and developed algorithms and tools for (1) automatically retrofitting binary legacy software with access controls, and (2) formally machine-certifying that the retrofitted software satisfies user-specified security policies. The research resulted in new software security systems for Java, ActionScript, and x86 native code that provably secure legacy code without any form of code-producer cooperation (e.g., source code or compiler support).

1 Summary of Achievements

1.1 Research Outcomes

Research supported by this contract resulted in the development of three major software security systems with associated discoveries and innovations. All publications and theses cited in this report are available for download from the following web page:

`http://www.utdallas.edu/~hamlen/research.html`

1. We developed the **Security Policy Xml (SPoX)** tool suite: the first fully declarative, aspect-oriented policy specification and in-lined reference monitor (IRM) system. SPoX includes tools for parsing, analyzing, and visualizing XML-based security policy specifications and

untrusted Java bytecode binaries. Design, implementation, and experimental results are detailed in the following publications and theses: [2, 9, 10, 11, 12, 13, 14, 18].

2. We discovered a new, more powerful IRM-certification paradigm based on model-checking. This was implemented in the Chekov✓ verification system, which automatically machine-verifies the policy-compliance of IRM-instrumented Java and ActionScript bytecode binaries. Design, implementation, and experimental results are detailed in the following publications and thesis: [1, 3, 4, 8, 9, 15, 16, 17].

3. We designed and implemented **Reins**: a new, machine-certified software fault isolation (SFI) system for native x86 architectures that implements IRMs for Intel-based Windows and Linux systems without any code-producer cooperation, such as compile-side support, source code, debug symbols, or online symbol stores. Its design and implementation are detailed in the following publications: [5, 19]. Two additional publications are submitted and currently under review.

1.2 Executive Summary of Conclusions

We met all four of the primary goals proposed for the project:

- Our ActionScript and x86 native code IRM implementations successfully incorporated machine-verifiable code optimizations during security retrofitting. This sufficed to offset much of the enforcement overhead. For x86 native code, we report overheads of less than 3%—substantially better than any prior system of equivalent capability to our knowledge [5].

- Our model-checking approach to IRM certification successfully verified dataflow-sensitive optimizations [4].

- SPoX facilitated formal policy analyses, such as policy inconsistency detection and elimination, that are provably undecidable with traditional, non-declarative aspect-oriented specification approaches [12].

- We successfully extended all of the above technologies to untyped, x86 native code software for real-world operating systems (Windows and Linux) [5].

We conclude that certified, in-lined reference monitoring is a highly feasible, flexible, and efficient approach to enforcing software security policies over binary legacy software. Additional applications of the technology are being explored in several subsequent projects, detailed in the next section.

1.3 Contribution to Other Awards and Contracts

The discoveries above have spawned three major ongoing research initiatives, currently supported by awards from the National Science Foundation (NSF), U.S. Army, and Air Force Office of Scientific Research (AFOSR):

Securing Web Advertisements (NSF, TC:Medium, $1.2M, 2011–2014). In collaboration with the University of Illinois at Chicago (UIC), we are applying our ActionScript certifying IRM system to develop security systems for mobile web advertisements. Malicious web ads (*malvertisements*) are a major ongoing concern for end users, publishers, ad distribution networks, and advertisers. Our ongoing work leverages the IRM technologies developed and reported here to provide provably sound and transparent protections for web ad domains.

Language-based Security for Polymorphic Malware Defense (NSF CAREER, TC, $500K, 2011–2016). Our successful extension of machine-certified SFI/IRM technologies to x86 native code architectures (see achievement 3 of §1.1) is a significant milestone toward extending powerful language-based security technologies to COTS native code architectures. Last year the PI received an NSF CAREER award for ongoing research that develops language-based protections for binary software that is potentially self-modifying, untyped, memory-unsafe, and obfuscated to resist disassembly.

Reactively Adaptive Malware (AFOSR, FA9550-10-1-0088, $450K, 2011–2014) (U.S. Army, $350K, 2011–2012). The binary analysis and transformation discoveries reported here are also being applied for active defense. Our ongoing *reactively adaptive malware* project develops mobile code that detects, adapts, and avoids antiviral defenses fully automatically in the wild. Such technologies are important for anticipating and understanding next-generation malware, and for counter-attacking cyber-attackers.

2 Educational Outcomes

2.1 Student Support

Funding from this award partially supported 5 graduate students:

- 4 Ph.D. students: Micah Jones (graduated December 2011 [9], now employed by L-3 Communications), Meera Sridhar, Vishwath Mohan, and Richard Wartell (expected graduations within the next 1.5 years); and

- 1 Masters student: Aditi Patwardhan (graduated June 2010 [14]).

Micah's thesis [9] developed the SPoX system (see outcome 1 of §1.1) and its support for the Chekov✓ verifier (see outcome 2 of §1.1). Aditi's thesis [14] developed a visualization system for SPoX and Java bytecode [13]. Meera's ongoing thesis work develped Chekov✓ and is extending the technology to transparency verification of web ad IRMs (see §1.3). Vishwath's and Richard's ongoing theses developed the Reins system (see outcome 3 of §1.1) and are continuing with its application to polymorphic malware defense and reactively adaptive malware (see §1.3).

2.2 Course Development

Research conducted under this contract contributed to the development of substantial educational material that augmented 3 different courses at UTD:

- CS6V81/7301: Language-based Security (Spring '08, Spring '11) [average student evaluation: 4.84 / 5 = Excellent];

- CS6371: Advanced Programming Languages (Fall '08, Spring '09, Fall '09, Spring '10, Spring '11) [average student evaluation: 4.21 / 5 = Very Good];

- CS4384: Automata Theory (Fall '10, Fall '11) [average student evaluation: 4.41 / 5 = Very Good]

CS6V81/7301: Language-based Security is a graduate-level elective that trained students in advanced software security technologies such as IRMs, SFI, information flow controls, malware analysis, and binary obfuscation.

Students received direct, hands-on experience with discoveries and tools resulting from this contract.

CS6371: Advanced Programming Languages is a grad-level core course that teaches language and compiler design. As a result of this contract, the course was significant augmented with examples and content motivated by secure software development and validation. Students learned type-theoretic and axiomatic semantical approaches to software security analysis.

CS4384: Automata Theory is an undergraduate core course that teaches formal languages and introductory computational complexity. The course was augmented with significant security content including automata-based approaches to security policy specification and analysis.

Federal CyberSecurity Scholarship For Service (NSF, $1.7M, 2010–2014). The educational developments above contributed to the establishment and enhancement of a new, NSF-supported Scholarship For Service (SFS) program at UTD in 2010, which recruits and trains undergraduates and graduates for federal cyber-security employment. The courses above have been instrumental for recruiting students into the program.

Publications

[1] Brian W. DeVries, Gopal Gupta, Kevin W. Hamlen, Scott Moore, and Meera Sridhar. ActionScript bytecode verification with co-logic programming. In Stephen Chong and David A. Naumann, editors, *Proceedings of the ACM SIGPLAN Workshop on Programming Languages and Analysis for Security (PLAS)*, pages 9–15, Dublin, Ireland, June 2009.

[2] Kevin W. Hamlen and Micah Jones. Aspect-oriented in-lined reference monitors. In Úlfar Erlingsson and Marco Pistoia, editors, *Proceedings of the ACM SIGPLAN Workshop on Programming Languages and Analysis for Security (PLAS)*, pages 11–20, Tucson, Arizona, June 2008.

[3] Kevin W. Hamlen, Micah M. Jones, and Meera Sridhar. Chekov: Aspect-oriented runtime monitor certification via model-checking. Technical Report UTDCS-16-11, Computer Science Department, The University of Texas at Dallas, Richardson, Texas, May 2011. `http://www.utdallas.edu/~hamlen/hamlen-utdcs-16-11.pdf`.

[4] Kevin W. Hamlen, Micah M. Jones, and Meera Sridhar. Aspect-oriented runtime monitor certification. In *Proceedings of the 18th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, Tallinn, Estonia, March–April 2012. forthcoming.

[5] Kevin W. Hamlen, Vishwath Mohan, and Richard Wartell. Reining in Windows API abuses with in-lined reference monitors. Technical Report UTDCS-18-10, Computer Science Department, The University of Texas at Dallas, Richardson, Texas, June 2010.

[6] Kevin W. Hamlen, Greg Morrisett, and Fred B. Schneider. Certified in-lined reference monitoring on .NET. In Vugranam C. Sreedhar and Steve Zdancewic, editors, *Proceedings of the 1st ACM SIGPLAN Workshop on Programming Languages and Analysis for Security (PLAS)*, pages 7–16, Ottawa, Ontario, June 2006.

[7] Kevin W. Hamlen, Greg Morrisett, and Fred B. Schneider. Computability classes for enforcement mechanisms. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 28(1):175–205, January 2006.

[8] Kevin W. Hamlen and Bhavani Thuraisingham. Secure semantic computing. *International Journal of Semantic Computing*, 5(2):121–131, June 2011.

[9] Micah Jones. *Declarative Aspect-oriented Security Policies for In-lined Reference Monitors*. PhD thesis, The University of Texas at Dallas, Richardson, Texas, December 2011. `http://www.utdallas.edu/~hamlen/jones11thesis.pdf`.

[10] Micah Jones and Kevin W. Hamlen. Enforcing IRM security policies: Two case studies. In *Proceedings of the IEEE Intelligence and Security Informatics Conference (ISI)*, pages 214–216, Dallas, Texas, June 2009.

[11] Micah Jones and Kevin W. Hamlen. A service-oriented approach to mobile code security. In Elhadi Shakshuka and Muhammad Younas, editors, *Proceedings of the 8th International Conference on Mobile Web Information Systems (MobiWIS)*, pages 531–538, Niagara Falls, Ontario, September 2011.

[12] Micah Jons and Kevin W. Hamlen. Disambiguating aspect-oriented security policies. In Jean-Marc Jézéquel and Mario Südholt, editors, *Proceedings of the 9th International Conference on Aspect-Oriented Software Development (AOSD)*, pages 193–204, Rennes, France, March 2010.

[13] Aditi Patwardhan, Kevin W. Hamlen, and Kendra Cooper. Towards security-aware program visualization for analyzing in-lined reference monitors. In *Proceedings of the International Workshop on Visual Languages and Computing (VLC)*, pages 257–260, Oak Brook, Illinois, October 2010.

[14] Aditi A. Patwardhan. Security-aware program visualization for analyzing in-lined reference monitors. Master's thesis, The University of Texas at Dallas, Richardson, Texas, June 2010. `http://www.utdallas.edu/~hamlen/patwardhan10thesis.pdf`.

[15] Meera Sridhar and Kevin W. Hamlen. ActionScript in-lined reference monitoring in Prolog. In Manuel Carro and Ricardo Peña, editors, *Proceedings of the 12th International Symposium on Practical Aspects of Declarative Languages (PADL)*, pages 149–151, Madrid, Spain, January 2010.

[16] Meera Sridhar and Kevin W. Hamlen. Model-checking in-lined reference monitors. In Gilles Barthe and Manuel V. Hermenegildo, editors, *In Proceedings of the 11th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI)*, pages 312–327, Madrid, Spain, January 2010.

[17] Meera Sridhar and Kevin W. Hamlen. Flexible in-lined reference monitor certification: Challenges and future directions. In Ranjit Jhala and Wouter Swierstra, editors, *Proceedings of the 5th ACM SIGPLAN Workshop on Programming Languages meets Program Verification (PLPV)*, pages 55–60, Austin, Texas, January 2011.

[18] Bhavani Thuraisingham and Kevin W. Hamlen. Challenges and future directions of software technology: Secure software development, invited paper. In Seikh Iqbal Ahamed, Doo-Hwan Bae, Sung Deok Cha, Carl K. Chang, Rajesh Subramanyan, Eric Wong, and Hen-I Yang, editors, *Proceedings of the 34th IEEE Annual International Computer Security and*

Applications Conference (COMPSAC), pages 17–20, Seoul, Korea, July 2010.

[19] Richard Wartell, Yan Zhou, Kevin W. Hamlen, Murat Kantarcioglu, and Bhavani Thuraisingham. Differentiating code from data in x86 binaries. In Dimitrios Gunopulos, Thomas Hofmann, Donato Malerba, and Michalis Vazirgiannis, editors, *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, volume 3, pages 522–536, 2011.